

RECEIVED
CENTRAL FAX CENTER

SEP 13 2005

**Yee &
Associates, P.C.**4100 Alpha Road
Suite 1100
Dallas, Texas 75244Main No. (972) 385-8777
Facsimile (972) 385-7766**Facsimile Cover Sheet**

To: Commissioner for Patents for Examiner Camquy Truong Group Art Unit 2127	Facsimile No.: 571/273-8300
From: Stephanie Fay for Carrie Parker Legal Assistant to Vicky Ash	No. of Pages Including Cover Sheet: 29
Message: Enclosed herewith: <ul style="list-style-type: none">• Transmittal Document; and• Appeal Brief.	
RECEIVED OIPE/IAP SEP 14 2005	
Re: Application No. 09/895,979 Attorney Docket No: AUS920010501US1	
Date: Tuesday, September 13, 2005	
Please contact us at (972) 385-8777 if you do not receive all pages indicated above or experience any difficulty in receiving this facsimile.	<i>This Facsimile is intended only for the use of the addressee and, if the addressee is a client or their agent, contains privileged and confidential information. If you are not the intended recipient of this facsimile, you have received this facsimile inadvertently and in error. Any review, dissemination, distribution, or copying is strictly prohibited. If you received this facsimile in error, please notify us by telephone and return the facsimile to us immediately.</i>

**PLEASE CONFIRM RECEIPT OF THIS TRANSMISSION BY
FAXING A CONFIRMATION TO 972-385-7766.**

RECEIVED
CENTRAL FAX CENTER

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

SEP 13 2005

In re application of: Shah et al.

Serial No.: 09/895,979

Filed: June 29, 2001

For: Methods and Apparatus in a
Logging System for the Tracking of
Tasks Solely Based on Function for
Data Analysis

35525

PATENT TRADEMARK OFFICE
CUSTOMER NUMBER§
§
§
§
§
§

Group Art Unit: 2127

Examiner: Truong, Camquy

Attorney Docket No.: AUS920010501US1

Certificate of Transmission Under 37 C.F.R. § 1.8(a)I hereby certify this correspondence is being transmitted via facsimile to
the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-
1450, facsimile number (571) 273-8300 on September 13, 2005.

By:


Stephanie FayTRANSMITTAL DOCUMENTCommissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

ENCLOSED HEREWITH:

- Appeal Brief (37 C.F.R. 41.37).

A fee of \$500.00 is required for filing an Appeal Brief. Please charge this fee to Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to Deposit Account No. 09-0447.

Respectfully submitted,


Gerald H. Glanzman

Registration No. 25,035

Duke W. Yee

Registration No. 34,285

YEE & ASSOCIATES, P.C.

P.O. Box 802333

Dallas, Texas 75380

(972) 385-8777

ATTORNEYS FOR APPLICANTS

RECEIVED
CENTRAL FAX CENTER

Docket No. AUS920010501US1

SEP 13 2005

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Shah et al.

Serial No. 09/895,979

Filed: June 29, 2001

For: Methods and Apparatus in a Logging System for the Tracking of Tasks Solely Based on Function for Data Analysis

[illegible]

Group Art Unit: 2127

Examiner: Truong, Camquy

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Certificate of Transmission Under 37 C.F.R. § 1.8(a)

I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (571) 273-8300 on September 13, 2005.

By:

Stephanie Fay

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on July 13, 2005.

The fees required under § 41.20(B)(2), and any required petition for extension of time for filing this brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

09/14/2005 SSITHIB1 00000005 090447 09895979
01 FC:1402 500.00 DA

(Appeal Brief Page 1 of 27)
Shah et al. - 09/895,979

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1-22

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: NONE
2. Claims withdrawn from consideration but not canceled: NONE
3. Claims pending: 1-22
4. Claims allowed: NONE
5. Claims rejected: 1-22
6. Claims objected to: NONE

C. CLAIMS ON APPEAL

The claims on appeal are: 1-22

STATUS OF AMENDMENTS

Amendments were made to claims 1, 11, and 21 in the Amendment After Final Office Action dated July 13, 2005 to address the rejection under 35 U.S.C. 112, second paragraph. As of this appeal brief, the status of the amendments is unknown. The amendments are included in the content of this appeal brief.

SUMMARY OF CLAIMED SUBJECT MATTER

A. CLAIM 1 - INDEPENDENT

The subject matter of claim 1 is directed to a method for tracking tasks in a logging system (400) (see *Abstract*, page 28, lines 5-15). A request is received at a log task manager (402) (see *Abstract*, page 28, lines 7-8). The request is associated with an application program (502, 504 and 506) to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked. The relationship between the set of related events and the task is established by the application program (see *Specification*, page 15, line 23 through page 16, line 4 and page 19, lines 15-17). The unique task identification is generated at a log task manager (see *Specification*, page 15, lines 9-22 and page 16, lines 5-18). The unique task identification is attached to a transport mechanism that passes information between components (see *Specification*, page 5, lines 8-10 and page 16, line 19 through page 17, line 16). The unique task identification is combined with logging information generated by one or more of the components to correlate logging information entries (438) associated with related events (see *Specification*, page 12, lines 5-14; page 17, line 6 through page 18, line 16; and page 19, lines 1-27). A plurality of logging information entries are filtered based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user (see *Specification*, page 14, line 11 through page 15, line 8 and page 18, line 17 through page 20, lines 10).

B. CLAIM 11 - INDEPENDENT

The subject matter of claim 11 is directed to a computer program product in a computer readable media for use in a data processing system (100, 200, 300) for tracking tasks in a logging system (400) (see *Abstract*, page 28, lines 5-15). The computer program product provides instructions for receiving a request at a log task manager (402) (see *Abstract*, page 28, lines 7-8). The request is associated with an application program (502, 504 and 506) to assign a unique task identification to a set of related events having a relationship with a task identified by the

application program to be tracked. The relationship between the set of related events and the task is established by the application program (see *Specification*, page 15, line 23 through page 16, line 4 and page 19, lines 15-17). The computer program product provides instructions for generating the unique task identification at a log task manager (see *Specification*, page 15, lines 9-22 and page 16, lines 5-18). The computer program product provides instructions for attaching the unique task identification to a transport mechanism that passes information between components (see *Specification*, page 5, lines 8-10 and page 16, line 19 through page 17, line 16).

The computer program product provides instructions for combining the unique task identification with logging information generated by one or more of the components to correlate logging information entries (438) associated with related events (see *Specification*, page 12, lines 5-14; page 17, line 6 through page 18, line 16; and page 19, lines 1-27). The computer program product provides instructions for filtering a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user (see *Specification*, page 14, line 11 through page 15, line 8 and page 18, line 17 through page 20, lines 10).

C. CLAIM 21 - INDEPENDENT

The subject matter of claim 21 is directed to a system for tracking tasks in a logging system (400) (see *Abstract*, page 28, lines 5-15). A logging manager (402) (see *Abstract*, page 28, lines 7-8) receives a request associated with an application program (502, 504 and 506) to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked. The relationship between the set of related events and the task is established by the application program (see *Specification*, page 15, line 23 through page 16, line 4 and page 19, lines 15-17). A unique taskID generator (404) generates the unique task identification (see *Specification*, page 15, lines 9-22 and page 16, lines 5-18). A task transport unit (406) attaches the unique task identification to a transport mechanism that passes information between components (see *Specification*, page 5, lines 8-10 and page 16, line 19 through page 17, line 16). A logger (415, 416, and 417) combines the unique task identification with logging information generated by one or more of the components to correlate logging information entries (438) associated with related events (see *Specification*, page 12, lines 5-14;

page 17, line 6 through page 18, line 16; and page 19, lines 1-27). A filter (420) filters a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user (see *Specification*, page 14, line 11 through page 15, line 8 and page 18, line 17 through page 20, lines 10).

D. CLAIM 2 - DEPENDENT

The subject matter of claim 2, which depends from claim 1, is directed to a method wherein attaching the unique task identification to the transport mechanism comprises attaching the unique task identification to a local thread transport (see *Specification*, page 5, lines 8-10 and page 16, line 19 through page 17, line 16).

E. CLAIM 3 - DEPENDENT

The subject matter of claim 3, which depends from dependent claim 2, is directed to a method further comprising extending the inheritable thread local at the local thread transport and placing the task identification on a local thread at the local thread transport (see *Specification*, page 16, line 19 through page 17, line 16).

F. CLAIM 12 - DEPENDENT

The subject matter of claim 12, which depends from claim 11, is directed to a computer program product wherein attaching the unique task identification to the transport mechanism comprises attaching the unique task identification to a local thread transport (see *Specification*, page 5, lines 8-10 and page 16, line 19 through page 17, line 16).

G. CLAIM 13 - DEPENDENT

The subject matter of claim 13, which depends from claim 12, is directed to a computer program product further providing instructions for extending the inheritable thread local at the local thread transport and instructions for placing the task identification on a local thread at the local thread transport (see *Specification*, page 16, line 19 through page 17, line 16).

H. CLAIM 9 - DEPENDENT

The subject matter of claim 9, which depends from claim 1, is directed to a method wherein the unique task identification is a first unique task identification and the related events are first related serial events. The log task manager (402) receives a request from the application program (502, 504 and 506) for a second unique task identification assigned to second related serial events identified by the application program. The second unique task identification is attached to the transport mechanism (see *Specification*, page 15, line 24 through page 18, line 15 and page 19, lines 11-26).

I. CLAIM 19 - DEPENDENT

The subject matter of claim 19, which depends from claim 11, is directed to a computer program product wherein the unique task identification is a first unique task identification and the related events are first related serial events. The computer program product provides instructions for receiving, at the log task manager (402), a request from the application program (502, 504 and 506) for a second unique task identification assigned to second related serial events identified by the application program. The computer program product provides instructions for attaching the second unique task identification to the transport mechanism (see *Specification*, page 15, line 24 through page 18, line 15 and page 19, lines 11-26).

GROUND OF REJECTION TO BE REVIEWED ON APPEAL**A. GROUND OF REJECTION 1 (Claims 1-22)**

The Final Office Action rejects claims 1-22 under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which Appellants regard as the invention.

B. GROUND OF REJECTION 2 (Claims 1-2, 4-12, and 14-22)

The Final Office Action rejects claims 1-2, 4-12, and 14-22 under 35 U.S.C. 103(a) as being allegedly unpatentable over Niemi et al. (U.S. 6,470,388 B1) in view of Teng et al. (U.S. 6,094,679).

C. GROUND OF REJECTION 3 (Claims 3 and 13)

The Final Office Action rejects claims 3 and 13 under 35 U.S.C. 103(a) as being allegedly unpatentable over Niemi et al. (U.S. 6,470,388 B1) in view of Teng et al. (U.S. 6,094,679), as applied in claims 1, 11, and 21, and further in view of Block (U.S. 6,820,261 B1).

ARGUMENT

A. GROUND OF REJECTION 1 (Claims 1-22)

The Final Office Action rejects claims 1-22 under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which Appellants regard as the invention. This rejection is respectfully traversed.

As to independent claims 1, 11, and 22, the Final Office Action states:

A. The claim language in the following claims is not clearly understood:

i. As to claims 1, 11 and 21, lines 4-5, it is not clearly understood how "set of related events" relates to a task (i.e. a task has a plurality of events?); line 6, it is not clearly indicated whether "the event" refers to "set of related events" in line 4.

Final Office Action dated April 13, 2005, page 1.

Claims 1, 11, and 21 were amended in the Amendment After Final Office Action dated July 13, 2005 to clarify that "the event" in line 6 refers to "the set of related events" as requested by the Examiner. The status of the proposed amendments is unknown as of this date, but Appellants have included these amendments in this appeal brief. Appellants respectfully submit that these proposed amendments overcome the rejection under 35 U.S.C. 112, second paragraph. Thus, Appellants respectfully request withdrawal of the rejection of claims 1-22 under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which Appellants regard as the invention.

B. GROUND OF REJECTION 2 (Claims 1-2, 4-12, and 14-22)

The Final Office Action rejects claims 1-2, 4-12, and 14-22 under 35 U.S.C. 103(a) as being allegedly unpatentable over Niemi et al. (U.S. 6,470,388 B1), hereinafter referred to as *Niemi*, in view of Teng et al. (U.S. 6,094,679), hereinafter referred to as *Teng*. This rejection is respectfully traversed.

B.1. Claims 1-2, 4-8, 10-12, 14-18, and 20-22

As to independent claims 1, 11, and 21, the Final Office Action states:

(Appeal Brief Page 11 of 27)
Shah et al. - 09/895,979

6. As to claims 1, 11 and 21, Niemi teaches the invention substantially as claimed including: A method for tracking in logging system (col. 3, lines 63-66), the method comprising:

Receiving, at log task manager (logging service layer, col. 4, line 13), a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by application program to be tracked, wherein the relationship between the events and the task is established by the application program (col. 4, lines 11-14; col. 6, lines 3-8; col. 8, lines 11-18; col. 10, lines 19-24; col. 16, lines 17-21 and lines 27-31);

Generating, at a log task manager, the unique task identification (col. 10, lines 19-24);

Combining the unique task identification with logging information generated by one of the components to correlate logging information entries associated with related events (col. 4, lines 29-32; col. 11, line 66-col. 12, line 15); and

Filtering a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user (Fig. 5; Fig. 6; col. 13, lines 16-40; col. 58-67).

7. Niemi does not explicitly teach attaching the unique task identification to a transport mechanism that passes information between components. However, Teng teaches an data field which provides the network server with information about the network client may be appended to HTTP formatted request message before issued to the network server (col. 7, lines 36-40).

8. It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Niemi and Teng because Teng's attaching the unique task identification to a transport mechanism that passes information between components would increase the flexibility of Niemi by providing attaching the unique task identification to a transport mechanism that passes information between components to minimize the risk that the network server return software files which are mismatched.

Final Office Action dated April 13, 2005, pages 2-3.

Claim 1, which is representative of the other rejected independent claims 11 and 21 with regard to similarly recited subject matter, reads as follows:

1. A method for tracking tasks in a logging system, the method comprising:
receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program;
generating, at a log task manager, the unique task identification;
attaching the unique task identification to a transport mechanism that passes information between components;

combining the unique task identification with logging information generated by one or more of the components to correlate logging information entries associated with related events; and

filtering a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user. (emphasis added)

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on the prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). For an invention to be *prima facie* obvious, the prior art must teach or suggest all claim limitations. *In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974).

Niemi and *Teng*, taken individually or in combination, do not teach or suggest "receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program" and "generating, at a log task manager, the unique task identification," as recited in claims 1, 11, and 21.

Niemi is directed to a system and method for centrally coordinating, collecting and storing error, trace, audit and other information in a computer network. Selected applications or processes running at various entities or hosts within the network instantiate one or more "debug" objects that collect particularized information. Each entity also includes at least one logging service layer that communicates with the application or process, and includes a communications resource and one or more state machine engines. In response to collecting error, trace, audit, or other information, each debug object passes it to the respective logging service layer, which decides whether or not to forward it to a centralized logging facility. The forwarding of collected information depends on the state of the debug object. If the state of the debug object is enabled, then the logging service layer forwards the collected information to the centralized logging facility. At the centralized logging facility, the information is time-stamped and appended to a log file along with the application's name and the name of the entity or host at which the application is running. See *Niemi*, Abstract. *Niemi* is cited for allegedly teaching the steps of claim 1 excluding attaching the unique task identification to a transport mechanism that passes information between components.

Niemi does not teach or suggest "receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program," as recited in claims 1, 11, and 21. Contrary to the Examiner's statements, the cited section of *Niemi* teaches that developers add specific code to their applications calling a debug object in response to the occurrence of some event or condition that may be of interest during debugging. The debug object collects and passes error, trace, audit or other information to the respective logging service layer. The logging serviced layer determines whether or not to forward the information to the centralized logging facility based on the state of the debug object.

Additionally, *Niemi* teaches that in response to constructing the first debug object, the logging service layer is initialized and creates a unique call-back for use in identifying a respective application. This is not teaching generating a unique task identification assigned to a set of related events having a relationship with a task identified by an application program to be tracked, as described in claims 1, 11, and 21. Thus, *Niemi* does not teach or suggest "generating, at a log task manager, the unique task identification," as recited in claims 1, 11, and 21.

The Final Office Action refers to the following portions of *Niemi* in the rejection of receiving step and the generating step of claims 1, 11 and 21:

Upon initialization, the selected applications or processes issue methods or calls to the respective logging service layer identifying their one or more debug objects.

Niemi, column 4, lines 11-14.

In addition, the application 208a running at workstation 202 includes a trap receiver process 306 and a client display process 308, while the application 208b running at workstation 204 includes a polling process 310 and a network topology process 312.

Niemi, column 6, lines 3-8.

In accordance with the present invention, the applications 208a, 208b and/or processes 306-312 are configured to construct and manipulate novel "debug" objects. FIG. 4 illustrates a simplified hierarchy of a base Debug class 400, which defines the generic behaviors of that class. The base Debug class 400 is preferably utilized to build a plurality of debug objects, one or more of which may be instantiated by applications 208a, 208b (FIG. 3) and/or processes 306-312.

Niemi, column 8, lines 11-18.

(Appeal Brief Page 14 of 27)
Shah et al. - 09/895,979

More specifically, in response to constructing the first debug object, the logging service layer (e.g., layer 316) for the respective application or process (e.g., trap receiver process 306) is initialized and accesses its call-back generator (e.g., generator 328) to create a unique call-back for use in identifying respective application or process.

Niemi, column 10, lines 19-24.

constructing one or more informational debug objects at a first application or process whereby each informational debug object corresponds to a particular type of error, trace, audit or other information generated by the first application or process;

Niemi, column 16, lines 17-21.

in response to obtaining error, trace, audit or other information at the first application or process, issuing a log command to the logging service layer that identifies the respective informational debug object and contains the obtained error, trace, audit or other information; and

Niemi, column 16, lines 27-31.

These portions of *Niemi* teach that developers add specific code to their applications calling a debug object in response to the occurrence of some event or condition that may be of interest during debugging. The debug object collects and passes error, trace, audit or other information to the respective logging service layer. In response to constructing the first debug object, the logging service layer is initialized and creates a unique call-back for use in identifying a respective application. The logging service layer determines whether or not to forward the information to the centralized logging facility based on the state of the debug object. *Niemi* does not teach or suggest "receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program" and "generating, at a log task manager, the unique task identification," as recited in claims 1, 11, and 21.

Teng is directed to a method of distributing software files resident on a network server to a network client. A world wide distributed printing solution is provided that is capable of working transparently on intranets and the Internet. *Teng* is cited for teaching a data field, which provides the network server with information about the network client, may be appended to a HTTP formatted request message before the message is issued to the network server. *Teng* does not teach or suggest "receiving, at log task manager, a request associated with an application

program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program," as recited in claims 1, 11, and 21. Further, like *Niemi*, *Teng* does not teach or suggest "generating, at a log task manager, the unique task identification," "a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked," or "wherein the relationship between the set of related events and the task is established by the application program," as recited in claims 1, 11, and 21.

Niemi and *Teng* fail to teach or suggest "receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program" and "generating, at a log task manager, the unique task identification." Therefore, the alleged combination of *Niemi* and *Teng* does not teach or suggest these features, as recited in independent claims 1, 11, and 21.

Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988); *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992).

One of ordinary skill in the art would not combine *Niemi* with *Teng* when the references are considered as a whole. In considering the references as a whole, one of ordinary skill in the art would take into account the problems recognized and solved. The present invention recognizes the problem of tracking a set of related events having a relationship with a task across multiple threads in a logging system and teaches away from a centralized storage repository of logging information (see Specification, page 11, line 30 through page 12, line 1; page 18, lines 9-15; and page 19, lines 11-26). *Niemi* and *Teng* do not teach the problem or its source. *Niemi* is directed to the problem of centralizing the storage of error, trace, audit and other information generated by distributed applications (see *Niemi*, column 3, lines 41-60). *Teng* is directed to a distributed printing solution that is capable of working transparently on intranets and the Internet (see *Teng*, Abstract, last 3 lines). One of ordinary skill in the art would therefore not be

motivated to combine or modify the references in the manner required to form the solution disclosed in the present invention.

Furthermore, as noted above, there is no teaching or suggestion in the references as to the desirability of including the features from the other references. *Niemi* does not mention a desirability for tracking a set of related events having a relationship with a task across multiple threads in a logging system. *Niemi* does not recognize the problem and is instead devoted to centralizing the disparate information generated by the custom debug objects. As the Examiner has failed to demonstrate any motivation or incentive in the prior art to combine and modify the references so as to achieve the claimed invention, the alleged combination can only be the result of impermissible hindsight reconstruction using Appellants' own disclosure as a guide. While Appellants understand that all examination entails some measure of hindsight, when the rejection is based completely on hindsight, as in the present case, to the exclusion of what can be gleaned from the references, then the rejection is improper and should be withdrawn.

In view of the above, Appellants respectfully request withdrawal of the rejection of claims 1, 11, and 21 under 35 U.S.C. § 103(a). Additionally, *Niemi* and *Teng*, taken individually or in combination, do not teach or suggest the features of dependent claims 2, 4-10, 12, 14-20, and 22 at least by virtue of their dependency on independent claims 1, 11, and 21, respectively. Therefore, Appellants respectfully request withdrawal of the rejection of claims 2, 4-10, 12, 14-20, and 22 under 35 U.S.C. § 103(a).

B.2. Claims 9 and 19

In addition to the above, Appellants respectfully submit that claims 9 and 19 are independently distinguishable from the *Niemi* and *Teng* references. Claim 9 depends from claim 1 and claim 19 depends from claim 11. Claims 9 and 19 additionally recite "wherein the unique task identification is a first unique task identification, the related events are first related serial events and further comprising: receiving, at the log task manager, a request from the application program for a second unique task identification assigned to second related serial events identified by the application program; and attaching the second unique task identification to the transport mechanism." As discussed above, *Niemi* and *Teng* do not teach or suggest the receiving step or the generating step of claims 1 and 11. Thus, *Niemi* and *Teng* do not teach or suggest receiving a

request from the application program for a second unique task identification assigned to second related serial events identified by the application program. Therefore, claims 9 and 19 are believed distinguished from the cited references. Appellants respectfully request withdrawal of the rejection of claims 9 and 19 under 35 U.S.C. § 103(a).

C. GROUND OF REJECTION 3 (Claims 3 and 13)

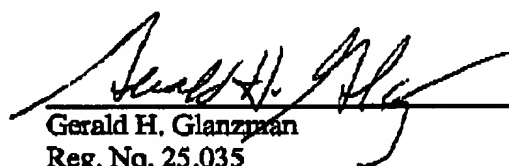
The Final Office Action rejects claims 3 and 13 under 35 U.S.C. 103(a) as being allegedly unpatentable over *Niemi* in view of *Teng*, as applied in claims 1, 11, and 21, and further in view of *Block* (U.S. 6,820,261 B1). This rejection is respectfully traversed.

C.1. Claims 3 and 13

Since claims 3 and 13 depend from independent claims 1 and 11, respectively, the same distinctions between *Niemi* and *Teng*, and the invention recited in claims 1 and 11, apply to dependent claims 3 and 13. Specifically, *Niemi* and *Teng*, taken individually or in combination, do not teach or suggest "receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program" and "generating, at a log task manager, the unique task identification." In addition, *Block* does not provide for the deficiencies of *Niemi* and *Teng* with regard to independent claims 1 and 11.

Block is directed to a system and method for providing automatic value inheritance when a parent thread creates a child thread. Upon thread creation, the system iterates over all of the inheritable thread-local values associated with a parent thread and initializes a child's value of these inheritable thread-local values, based on an appropriate child value method. The child's values may be a copy of the parent's values, or a predetermined function of the parent's values. See *Block*, Abstract. *Block* is cited for allegedly teaching the feature of extending the inheritable thread local at the local thread transport. *Block* does not teach or suggest the feature of "receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events

and the task is established by the application program" and "generating, at a log task manager, the unique task identification," as recited in claims 1 and 11. Thus, any alleged combination of *Block* with *Niemi* and *Teng* still would not result in the invention recited in claims 1 and 11 from which claims 3 and 13 depend. Accordingly, Appellants respectfully request withdrawal of the rejection of claims 3 and 13 under 35 U.S.C. § 103(a).



Gerald H. Glanzman
Reg. No. 25,035
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

GHG/vja

CLAIMS APPENDIX

The text of the claims involved in the appeal are:

1. A method for tracking tasks in a logging system, the method comprising:

receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program;

generating, at a log task manager, the unique task identification;

attaching the unique task identification to a transport mechanism that passes information between components;

combining the unique task identification with logging information generated by one or more of the components to correlate logging information entries associated with related events; and

filtering a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user.

2. The method as recited in claim 1, wherein attaching the unique task identification to the transport mechanism comprises attaching the unique task identification to a local thread transport.

3. The method as recited in claim 2, further comprising:
at the local thread transport, extending the inheritable thread local; and
at the local thread transport, placing the task identification on a local thread.
4. The method as recited in claim 1, wherein the transport mechanism utilizes a remote proxy call.
5. The method as recited in claim 1, wherein the transport mechanism utilizes port hardware.
6. The method as recited in claim 1, wherein the transport mechanism utilizes a point to point protocol.
7. The method as recited in claim 1, wherein the point to point protocol is a hypertext transfer protocol.
8. The method as recited in claim 1, wherein the transport mechanism utilizes a message context.
9. The method as recited in claim 1, wherein the unique task identification is a first unique task identification, the related events are first related serial events and further comprising:

receiving, at the log task manager, a request from the application program for a second unique task identification assigned to second related serial events identified by the application program; and

attaching the second unique task identification to the transport mechanism.

10. The method as recited in claim 1, further comprising:

mapping a taskID to a corresponding action, wherein the corresponding action provides a user friendly description of the related events; and

presenting logging information to a user based on the corresponding action.

11. A computer program product in a computer readable media for use in a data processing system for tracking tasks in a logging system, the computer program product comprising:

first instructions for receiving, at log task manager, a request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program;

second instructions for generating, at a log task manager, the unique task identification;

third instructions for attaching the unique task identification to a transport mechanism that passes information between components;

fourth instructions for combining the unique task identification with logging information generated by one or more of the components to correlate logging information entries associated with related events; and

fifth instructions for filtering a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user.

12. The computer program product as recited in claim 11, wherein attaching the unique task identification to the transport mechanism comprises attaching the unique task identification to a local thread transport.

13. The computer program product as recited in claim 12, further comprising:
sixth instructions, at the local thread transport, for extending the inheritable thread local;
and

seventh instruction, at the local thread transport, for placing the task identification on a local thread.

14. The computer program product as recited in claim 11, wherein the transport mechanism utilizes a remote proxy call.

15. The computer program product as recited in claim 11, wherein the transport mechanism utilizes port hardware.

16. The computer program product as recited in claim 11, wherein the transport mechanism utilizes a point to point protocol.

17. The computer program product as recited in claim 11, wherein the point to point protocol is a hypertext transfer protocol.

18. The computer program product as recited in claim 11, wherein the transport mechanism utilizes a message context.

19. The computer program product as recited in claim 11, wherein the unique task identification is a first unique task identification, the related events are first related serial events and further comprising:

sixth instructions for receiving, at the log task manager, a request from the application program for a second unique task identification assigned to second related serial events identified by the application program; and

seventh instructions for attaching the second unique task identification to the transport mechanism.

20. The computer program product as recited in claim 11, further comprising:

sixth instructions for mapping a taskID to a corresponding action, wherein the corresponding action provides a user friendly description of the related events; and

seventh instructions for presenting logging information to a user based on the corresponding action.

21. A system for tracking tasks in a logging system, the computer program product comprising:

a logging manager which receives request associated with an application program to assign a unique task identification to a set of related events having a relationship with a task identified by the application program to be tracked, wherein the relationship between the set of related events and the task is established by the application program;

a unique taskID generator which generates the unique task identification;

a task transport unit which attaches the unique task identification to a transport mechanism that passes information between components;

a logger which combining the unique task identification with logging information generated by one or more of the components to correlate logging information entries associated with related events; and

a filter which filters a plurality of logging information entries based on the unique task identification to produce a set of correlated logging information entries associated with the related events for presentation to a user.

22. The computer program product as recited in claim 11, further comprising:

a mapper which maps a taskID to a corresponding action, wherein the corresponding action provides a user friendly description of the related events; and

a presentation unit which presents logging information to a user based on the corresponding action.

EVIDENCE APPENDIX

There is no evidence to be presented.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings.